



# UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE  
United States Patent and Trademark Office  
Address: COMMISSIONER FOR PATENTS  
P.O. Box 1450  
Alexandria, Virginia 22313-1450  
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/801,589	03/08/2001	David H. Evans	AUS920010023US1	4875

7590 02/19/2004

Robert H. Frantz  
P.O. Box 23324  
Oklahoma, OK 73123-2334

EXAMINER
----------

TANG, KUO LIANG J

ART UNIT	PAPER NUMBER
----------	--------------

2122

DATE MAILED: 02/19/2004

3

Please find below and/or attached an Office communication concerning this application or proceeding.

## Office Action Summary

Application No.

09/801,589

Applicant(s)

EVANS ET AL.

Examiner

Kuo-Liang J Tang

Art Unit

2122

-- Th MAILING DATE of this communication appears on the cover sheet with the correspondence address --

### Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

### Status

- 1) ☒ Responsive to communication(s) filed on 08 March 2001.
- 2a) ☐ This action is **FINAL**. 2b) ☒ This action is non-final.
- 3) ☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

### Disposition of Claims

- 4) ☒ Claim(s) 1-37 is/are pending in the application.
- 4a) Of the above claim(s) \_\_\_\_\_ is/are withdrawn from consideration.
- 5) ☐ Claim(s) \_\_\_\_\_ is/are allowed.
- 6) ☒ Claim(s) 1-37 is/are rejected.
- 7) ☐ Claim(s) \_\_\_\_\_ is/are objected to.
- 8) ☐ Claim(s) \_\_\_\_\_ are subject to restriction and/or election requirement.

### Application Papers

- 9) ☐ The specification is objected to by the Examiner.
- 10) ☐ The drawing(s) filed on \_\_\_\_\_ is/are: a) ☐ accepted or b) ☐ objected to by the Examiner.  
Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).  
Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).
- 11) ☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

### Priority under 35 U.S.C. § 119

- 12) ☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).
- a) ☐ All b) ☐ Some \* c) ☐ None of:
1. ☐ Certified copies of the priority documents have been received.
2. ☐ Certified copies of the priority documents have been received in Application No. \_\_\_\_\_.
3. ☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

\* See the attached detailed Office action for a list of the certified copies not received.

### Attachment(s)

- 1) ☒ Notice of References Cited (PTO-892)
- 2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
- 3) ☒ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08)  
Paper No(s)/Mail Date 2.
- 4) ☐ Interview Summary (PTO-413)  
Paper No(s)/Mail Date. \_\_\_\_\_.
- 5) ☐ Notice of Informal Patent Application (PTO-152)
- 6) ☐ Other: \_\_\_\_\_.

### **DETAILED ACTION**

1. Claims 1-37 are pending and have been examined. The priority date for this application is 03/08/2001.

### ***Specification***

2. The specification of the disclosure is objected to because

Pg. 6, Ln 13; Pg. 7, Ln 3, 6; Pg. 8, Ln 13, 17, 21; Pg. 10, Ln 5; Pg 11, Ln 17; Pg. 40, Ln 5; "JDPA" should be "JPDA". Correction is required. See MPEP § 608.01(b).

### ***Claim Rejections - 35 USC § 103***

The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

(a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negated by the manner in which the invention was made.

3. Claims 1-2, 4, 6-10, 12-15, 17, 19-23, 25-28, 31-34 and 36-37 are rejected under 35 U.S.C. 103(a) as being unpatentable over Edwards et al. US. Patent No. 5,901,315 in view of SUN Inc, "Java Platform Debugger Architecture – Documentation JPDA 1.0", (hereinafter SUNJPDA).

As Per Claim 1, Edwards teaches a method for debugging a target application comprising Java code having native method dll's associated therewith. The method is carried out in a computer having an operating system, a system debug application programming interface (API),

Art Unit: 2122

and a Java virtual machine having a Java debug API. (E.g. see Abstract and associated text). In that Edwards discloses the method that covering the steps of:

“launching a Java virtual machine under the system debug API,” (Eg. See Abstract);  
“executing the application under the Java virtual machine,” (Eg. See Abstract) and  
“simultaneous control of the target application via the system debug API and the Java debug API is enabled. Thus, the method effects the debug of the target application by simultaneously debugging the Java code and the native method dynamic load libraries.” (Eg. See Abstract).

Edwards does not explicitly disclose the Java code is debugged using the Java Platform Debugger Architecture Virtual Machine Debug Interface API. However, SUNJPDA teaches a debug tool which contain three parts (Java Virtual Machine Debugger Interface (JVMDI), Java Debug Wire Protocol (JDWP) and Java Debugger Interface (JDI)) to be used in Java environment. Therefore, it would have been obvious for one of ordinary skill in the art at the time the invention was made to combine the SUNJPDA debug tool into Edwards teaching because one would want to take advantages of known tool for such a particular environment, that is Java environment.

As per Claim 2, the rejection of claim 1 is incorporated and further Edwards teaches  
“providing and displaying a console message window which permits a user to view output printed by the application.” (E.g. see FIG. 2 blk 21 and col. 3:62-67 to 4:1-14, GUI);

As per Claim 4, the rejection of claim 1 is incorporated and further Edwards teaches

“providing and displaying a console message window which permits a user to write data to the application.” (E.g. see FIG. 2 blk 21 and col. 3:62-67 to 4:1-14, GUI);

As per Claim 6, the rejection of claim 1 is incorporated and further Edwards does not explicitly disclose “providing the ability to exclude classes when stepping execution by use of an asterisk when invoking an "addClassExclusionFilter" method on a step request object.” However, SUNJPDA teaches “addClassExclusionFilter” method on a step request object. (E.g. see SUNJPDA, pg. 52). Therefore, it would have been obvious for one of ordinary skill in the art at the time the invention was made to also want to take advantages of the known method on a step request object in Java environment.

As per Claim 7, the rejection of claim 1 is incorporated and further Edwards does not explicitly disclose “providing string evaluation with a StringReference interface which extends an ObjectReference such that a string is treated as a class.” However, SUNJPDA teaches StringReference (E.g. see SUNJPDA, pg. 73) and ObjectReference. (E.g. see SUNJPDA, pg. 70). Therefore, it would have been obvious for one of ordinary skill in the art at the time the invention was made to also want to take advantages of the known method on a step request object in Java environment.

As per Claim 8, the rejection of claim 1 is incorporated and further Edwards does not explicitly disclose “invoking a mirrorOf utility to obtain an access reference to a type of symbol to be modified; and invoking a setValue utility to modify the value of the symbol.” However,

Art Unit: 2122

SUNJPDA teaches mirrorOf utility (E.g. see SUNJPDA, pg. 68) and setValue utility (E.g. see SUNJPDA, pg. 71-72). Therefore, it would have been obvious for one of ordinary skill in the art at the time the invention was made to also want to take advantages of the known method on a step request object in Java environment.

As per Claim 9, the rejection of claim 1 is incorporated and further Edwards does not explicitly disclose obtaining a list of the loaded classes; obtaining line number information from a class object; instantiating a breakpoint request; setting the breakpoint request suspend policy; and enabling the breakpoint request. However, SUNJPDA teaches “obtaining a list of the loaded classes;” (E.g. see SUNJPDA, pg. 58, Interface:ClassLoaderReference, Method: definedClasses); “obtaining line number information from a class object;” (E.g. see SUNJPDA, pg. 67, Interface:ReferenceType, Method: locationsOfLine); “instantiating a breakpoint request;” (E.g. see SUNJPDA, pg. 55, Interface:BreakpointRequest, Method: BreakpointRequests); “setting the breakpoint request suspend policy;” (E.g. see SUNJPDA, pg. 71, Interface:EventRequest, Method:setSuspendPolicy); and “enabling the breakpoint request.” (E.g. see SUNJPDA, pg. 59, Interface: EventRequest, Method:enable). Therefore, it would have been obvious for one of ordinary skill in the art at the time the invention was made to also want to take advantages of the known method on a step request object in Java environment.

As per Claim 10, the rejection of claim 9 is incorporated and further Edwards does not explicitly disclose adding the breakpoint to a deferred breakpoint table if the class where the breakpoint is to be located is not already loaded. However, SUNJPDA teaches “adding the

Art Unit: 2122

breakpoint to a deferred breakpoint table if the class where the breakpoint is to be located is not already loaded.” (E.g. see SUNJPDA, pg. 78, Interface: WatchpointRequest, Method: addClassExclusionFilter). Therefore, it would have been obvious for one of ordinary skill in the art at the time the invention was made to also want to take advantages of the known method on a step request object in Java environment.

As per Claim 12, the rejection of claim 1 is incorporated and further Edwards does not explicitly disclose obtaining an event request manager from the virtual machine; invoking a method of the event request manager to create a new step request including a step size parameter; enabling the new step request; and resuming the virtual machine. However, SUNJPDA teaches “obtaining an event request manager from the virtual machine;” (E.g. see SUNJPDA, pg. 60, Interface: EventRequestManager, Method: EventRequestManager); “invoking a method of the event request manager to create a new step request including a step size parameter;” (E.g. see SUNJPDA, pg. 72, Interface: EventRequestManager, Method: createStepRequest); “enabling the new step request;” (E.g. see SUNJPDA, pg. 72, Interface: StepRequest, Method: size); and “resuming the virtual machine. “(E.g. see SUNJPDA, pg. 71, Interface: VirtualMachine, Method: resume). Therefore, it would have been obvious for one of ordinary skill in the art at the time the invention was made to also want to take advantages of the known method on a step request object in Java environment.

As per Claim 13, the rejection of claim 1 is incorporated and further Edwards does not explicitly disclose providing a plurality of event handlers, one event handler for each requested

Art Unit: 2122

event object to be processed; polling a virtual machine event queue to determine if any requested event objects are queued; removing requested event objects from the event queue; determining the event object type; and calling an event handler according to the determined event object type. However, SUNJPDA teaches “providing a plurality of event handlers, one event handler for each requested event object to be processed,” (E.g. see SUNJPDA, pg. 38, JVMDI, SetEventHook); “polling a virtual machine event queue to determine if any requested event objects are queued,” (E.g. see SUNJPDA, pg. 60, Interface: VirtualMachine, Method: eventQueue); “removing requested event objects from the event queue,” (E.g. see SUNJPDA, pg. 71, Interface: EventQueue, Method:remove); “determining the event object type,” (E.g. see SUNJPDA, pg. 70, Interface: ClassPrepareEvent, Method:referenceType); and “calling an event handler according to the determined event object type.” (E.g. see SUNJPDA, pg. 65, Interface: ObjectReference, Method: invokeMethod). Therefore, it would have been obvious for one of ordinary skill in the art at the time the invention was made to also want to take advantages of the known method on a step request object in Java environment.

As per Claim 14, is the computer-readable medium claim corresponding to the method claim 1 and is rejected under the same reason set forth in connection of the rejection of claim 1. Further Edwards discloses computer-readable medium (E.g. see col. 10:47-67 to col.12)



Art Unit: 2122

As per Claims 15, 17, 19-22, 25-26, the rejections of claim 14 are incorporated and are rejected under the same reason set forth in connection of the rejection of claims 2, 4, 6-9, 12-13 respectively.

As per Claim 23, the rejections of claim 22 are incorporated and are rejected under the same reason set forth in connection of the rejection of claim 10.

As Per Claim 27, Edwards teaches:

“a processor;” (E.g. see FIG. 2, blk 12);

“an operating system (E.g. see FIG. 2, blk 14); having a system debug application programming interface (API);” (E.g. see FIG. 2, blk 16);

“a Java interpreter for running an application program under a Java Virtual Machine;” (E.g. see col. 9:24-38);

“a Java probe means for accessing and controlling an application program being executed by said Java Virtual Machine via said Java API;” (E.g. see FIG. 2, blk 12); and

“a debugger engine daemon (E.g. see FIG. 3, blk 34), communicative to said Java probe (E.g. see FIG. 3, blk 36), for simultaneously debugging the Java code and native method dynamic load library code in coordination with said probe means.” (E.g. see Abstract).

Edwards does not explicitly disclose the Java code is debugged using the Java Platform Debugger Architecture Virtual Machine Debug Interface API. However, SUNJPDA teaches a debug tool which contain three parts (Java Virtual Machine Debugger Interface (JVMDI), Java Debug Wire Protocol (JDWP) and Java Debugger Interface (JDI)) to be used in Java environment.

Art Unit: 2122

Therefore, it would have been obvious for one of ordinary skill in the art at the time the invention was made to combine the SUNJPDA debug tool into Edwards teaching because one would want to take advantages of known tool for such a particular environment, that is Java environment.

As per Claim 28, the rejection of claim 27 is incorporated and further Edwards teaches “the debugger engine daemon is adapted to debug native method dynamic load libraries of the application under the system debug API and the Java probe is adapted to debug Java application code” (E.g. see Abstract).

Edwards does not explicitly disclose the Java code is debugged using the Java Platform Debugger Architecture Virtual Machine Debug Interface API. However, SUNJPDA teaches a debug tool which contain three parts (Java Virtual Machine Debugger Interface (JVMDI), Java Debug Wire Protocol (JDWP) and Java Debugger Interface (JDI)) to be used in Java environment. Therefore, it would have been obvious for one of ordinary skill in the art at the time the invention was made to also want to take advantages of the known method on a step request object in Java environment.

As per Claims 31-34, 36-37, the rejections of claim 27 are incorporated and are rejected under the same reason set forth in connection of the rejection of claims 6-9, 12-13 respectively.

Art Unit: 2122

4. Claims 3, 5, 11, 16, 18, 24, 29-30 and 35 are rejected under 35 U.S.C. 103(a) as being unpatentable over Edwards in view of SUNJPDA further in view of "Java 2 Platform, Standard Edition, v1.3.1 API specification, Index B, G, I, O", (hereinafter SUNJ2SEAPI).

As per Claim 3, the rejection of claim 2 is incorporated and further Edwards and SUNJPDA teaches

"obtaining a Process object from the virtual machine;" (E.g. see Edwards, FIG. 5A, item 72, and col. 7:66-67 to 7:1-62);

Edwards and SUNJPDA do not explicitly disclose obtaining an InputStream object from the Process object, creating a "BufferedReader" object; and using a read method to retrieve data from the application. However, Edwards teaches communication using socket (E.g. FIG. 5A), hence implicitly disclosing Java code interface handling socket stream or I/O stream.

BufferedReader (E.g. see SUNJ2SEAPI, Index B) and InputStream (E.g. see SUNJ2SEAPI, Index I) are included in the SUNJ2SEAPI. Therefore, it would have been obvious for one of ordinary skill in the art at the time the invention was made to include these BufferedReader and InputStream APIs into the system of Edwards. The modification would have been obvious because one of ordinary skill in the art would have been motivated to take advantages of the known APIs in a Java environment.

Art Unit: 2122

As per Claim 5, the rejection of claim 4 is incorporated and further Edwards and SUNJPDA teaches

“obtaining a Process object from the virtual machine;” (E.g. see Edwards, FIG. 5A, item 72, and col. 7:66-67 to 7:1-62);

Edwards and SUNJPDA do not explicitly disclose obtaining an OutputStream object from the Process object, creating a "BufferedWriter" object; and using a read method to retrieve data from the application. However, Edwards teaches communication using socket (E.g. FIG. 5A), hence implicitly disclosing Java code interface handling socket stream or I/O stream. BufferedWriter (E.g. see SUNJ2SEAPI, Index B) and OutputStream (E.g. see SUNJ2SEAPI, Index O) are included in the SUNJ2SEAPI. Therefore, it would have been obvious for one of ordinary skill in the art at the time the invention was made to include these BufferedWriter and OutputStream APIs into the system of Edwards. The modification would have been obvious because one of ordinary skill in the art would have been motivated to take advantages of the known APIS in a Java environment.

As per Claim 11, the rejection of claim 1 is incorporated and further Edwards and SUNJPDA teaches

Edwards and SUNJPDA do not explicitly disclose obtaining a reference to a method "getClassLoader"; and invoking said method to load a class. However, SUNJPDA teaches "getClassLoader" method (E.g. see SUNJ2SEAPI, Index G). Therefore, it would have been obvious for one of ordinary skill in the art at the time the invention was made to take advantages of the known method to load and invoke a class, that is Java.

Art Unit: 2122

As per Claim 16, the rejections of claim 15 are incorporated and are rejected under the same reason set forth in connection of the rejection of claim 3.

As per Claim 18, the rejections of claim 17 are incorporated and are rejected under the same reason set forth in connection of the rejection of claim 5.

As per Claim 24, the rejections of claim 14 are incorporated and are rejected under the same reason set forth in connection of the rejection of claim 11.

As per Claims 29-30, 35 the rejections of claim 27 are incorporated and are rejected under the same reason set forth in connection of the rejection of claims 3, 5, 10 respectively.

### ***Conclusion***

5. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Kuo-Liang J Tang whose telephone number is 703-305-4866.

The examiner can normally be reached on M-F 8:30 to 5:00.

***If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Tuan Q Dam can be reached on 703-305-4552.***

Art Unit: 2122

Any response to this action should be mailed to:

Commissioner of Patents and Trademarks

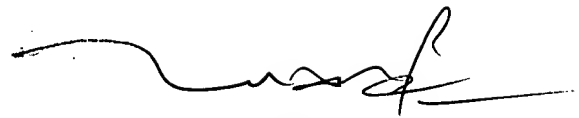
Washington, D.C. 20231

or faxed to:

(703) 872-9306.

*Kuo-Liang J. Tang*

Software Engineer Patent Examiner

A handwritten signature in black ink, appearing to read 'TUAN DAM', with a long horizontal flourish extending to the right.

**TUAN DAM  
SUPERVISORY PATENT EXAMINER**